Roxiler Systems Assignment

Problem Statement:

Tech Stack
● Backend: Any one of backend framework from this ExpressJs/Loopback/NestJs
● Database: PostgreSQL/MySQL
● Frontend: ReactJs
Requirements We need a web application that allows users to submit ratings for stores registered on the platform. The ratings should range from 1 to 5. A single login system should be implemented for all users. Based on their roles, users will have access to different functionalities upon logging in. Normal users should be able to sign up on the platform through a registration page. User Roles 1. System Administrator 2. Normal User 3. Store Owner
Functionalities System Administrator
● Can add new stores, normal users, and admin users.
● Has access to a dashboard displaying:
○ Total number of users
○ Total number of stores
○ Total number of submitted ratings
● Can add new users with the following details:
○ Name ○ Email ○ Password ○ Address
● Can view a list of stores with the following details:
○ Name, Email, Address, Rating
● Can view a list of normal and admin users with:
○ Name, Email, Address, Role
● Can apply filters on all listings based on Name, Email, Address, and Role.
● Can view details of all users, including Name, Email, Address, and Role. ○ If the user is a Store Owner, their Rating should also be displayed.
● Can log out from the system. Normal User
● Can sign up and log in to the platform.
● Signup form fields:
○ Name ○ Email ○ Address ○ Password
● Can update their password after logging in.
● Can view a list of all registered stores.
● Can search for stores by Name and Address.
● Store listings should display:
○ Store Name ○ Address ○ Overall Rating ○ User's Submitted Rating ○ Option to submit a rating ○ Option to modify their submitted rating
● Can submit ratings (between 1 to 5) for individual stores.
● Can log out from the system. Store Owner
● Can log in to the platform.
● Can update their password after logging in.
● Dashboard functionalities:
○ View a list of users who have submitted ratings for their store. ○ See the average rating of their store.
● Can log out from the system. Form Validations
● Name: Min 20 characters, Max 60 characters.
● Address: Max 400 characters.
● Password: 8-16 characters, must include at least one uppercase letter and one special character.
● Email: Must follow standard email validation rules.

Frontend Implementation:

Components(folder):

Profile.js:

```
import { useNavigate } from "react-router-dom";
```

```
function Profile() {
  const navigate = useNavigate();

  const navigateToUpdatePassword = () => {
    navigate("/update-password");
  };

  return (
    <div>
      <h2>User Profile</h2>
      <button onClick={navigateToUpdatePassword}>Update Password</button>
    </div>
  );
}

export default Profile;
```

UpdatePassword.js:

```
import React, { useState } from "react";
import styled from "styled-components";

const UpdatePasswordContainer = styled.div`
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  height: 100vh;
`;

const Input = styled.input`
  padding: 10px;
  margin: 10px;
  width: 250px;
`;

const Button = styled.button`
  padding: 10px;
  width: 270px;
  background-color: #007bff;
  color: white;
  cursor: pointer;
`;

function UpdatePassword() {
  const [currentPassword, setCurrentPassword] = useState("");
  const [newPassword, setNewPassword] = useState("");
  const [confirmPassword, setConfirmPassword] = useState("");

  const handleUpdatePassword = () => {
    if (newPassword === confirmPassword) {
      alert("Password updated successfully");
    } else {
      alert("Passwords do not match.");
    }
  };

  return (
    <UpdatePasswordContainer>
      <h2>Update Password</h2>
```

```
      <Input
        type="password"
        placeholder="Current Password"
        onChange={(e) => setCurrentPassword(e.target.value)}
      />
      <Input
        type="password"
        placeholder="New Password"
        onChange={(e) => setNewPassword(e.target.value)}
      />
      <Input
        type="password"
        placeholder="Confirm Password"
        onChange={(e) => setConfirmPassword(e.target.value)}
      />
      <Button onClick={handleUpdatePassword}>Update Password</Button>
    </UpdatePasswordContainer>
  );
}

export default UpdatePassword;
```

Pages(folder):

Dashboard.js:

```
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import styled from "styled-components";
import axios from "axios";

const Container = styled.div`
  max-width: 600px;
  margin: 50px auto;
  padding: 20px;
  background: white;
  border-radius: 10px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
  text-align: center;
`;

const Title = styled.h2`
  margin-bottom: 20px;
`;

const Input = styled.input`
  width: 90%;
  padding: 10px;
  margin: 10px 0;
  border: 1px solid #ccc;
  border-radius: 5px;
`;

const Button = styled.button`
  padding: 10px 15px;
  margin-top: 10px;
  background: #16a085;
  color: white;
  border: none;
  cursor: pointer;
```

```
  border-radius: 5px;
  transition: 0.3s;

  &:hover {
    background: #13876c;
  }
`;

const StoreList = styled.ul`
  list-style: none;
  padding: 0;
`;

const StoreItem = styled.li`
  background: #f8f8f8;
  margin: 10px 0;
  padding: 10px;
  border-radius: 5px;
`;

const Stars = styled.div`
  display: flex;
  justify-content: center;
  gap: 5px;
  margin-top: 5px;
`;

const Star = styled.span`
  font-size: 20px;
  cursor: pointer;
  color: ${({ active }) => (active ? "#f39c12" : "#ccc")};
`;

function Dashboard() {
  const navigate = useNavigate();
  const [stores, setStores] = useState([]);
  const [storeName, setStoreName] = useState("");
  const [storeAddress, setStoreAddress] = useState("");

  useEffect(() => {
    const savedStores = JSON.parse(localStorage.getItem("stores")) || [];
    setStores(savedStores);
  }, []);

  const saveStores = (updatedStores) => {
    localStorage.setItem("stores", JSON.stringify(updatedStores));
    setStores(updatedStores);
  };

  const addStore = () => {
    if (storeName.length < 3 || storeAddress.length < 5) {
      alert("Store Name must be at least 3 characters & Address at least 5.");
      return;
    }

    const newStore = { name: storeName, address: storeAddress, rating: 0 };
    saveStores([...stores, newStore]);
    setStoreName("");
    setStoreAddress("");
  };
```

```
const rateStore = (index, rating) => {
  const updatedStores = [...stores];
  updatedStores[index].rating = rating;
  saveStores(updatedStores);
};

const handleLogout = () => {
  localStorage.removeItem("isAuthenticated");
  localStorage.removeItem("userRole");
  navigate("/login");
};
const Dashboard = () => {
  const [stats, setStats] = useState({ users: 0, stores: 0, ratings: 0 });

  useEffect(() => {
    fetchStats();
  }, []);

  const fetchStats = async () => {
    try {
      const response = await axios.get("/api/admin/stats");
      setStats(response.data);
    } catch (error) {
      console.error("Error fetching stats", error);
    }
  };

  return (
    <div>
      <h2>Admin Dashboard</h2>
      <div>
        <p>Total Users: {stats.users}</p>
        <p>Total Stores: {stats.stores}</p>
        <p>Total Ratings: {stats.ratings}</p>
      </div>
    </div>
  );
};

return (
  <Container>
    <Title>Add Store & Rate</Title>
    <Input
      type="text"
      placeholder="Store Name"
      value={storeName}
      onChange={(e) => setStoreName(e.target.value)}
    />
    <Input
      type="text"
      placeholder="Store Address"
      value={storeAddress}
      onChange={(e) => setStoreAddress(e.target.value)}
    />
    <Button onClick={addStore}>Add Store</Button>

    <h3>Stores</h3>
    <StoreList>
      {stores.map((store, index) => (
```

```
        <StoreItem key={index}>
          <strong>{store.name}</strong> - {store.address}
          <Stars>
            {[1, 2, 3, 4, 5].map((star) => (
              <Star
                key={star}
                active={star <= store.rating}
                onClick={() => rateStore(index, star)}
              >
                ★
              </Star>
            ))}
          </Stars>
        </StoreItem>
      ))}
    </StoreList>
    <Button onClick={handleLogout}>Logout</Button>
  </Container>
 );
}

export default Dashboard;
```

Home.js:

```
import React from "react";
import { Link } from "react-router-dom";
import styled from "styled-components";

const HomeContainer = styled.div`
 display: flex;
 flex-direction: column;
 justify-content: center;
 align-items: center;
 height: 100vh;
 text-align: center;
`;

const Button = styled(Link)`
 background: linear-gradient(90deg, #ff6b6b, #ff8e53);
 color: white;
 padding: 12px 20px;
 border-radius: 5px;
 text-decoration: none;
 font-size: 18px;
 margin-top: 20px;
`;

function Home() {
 return (
  <HomeContainer>
   <h1 className="glow">Welcome to Store Rating Platform</h1>
   <Button to="/login">Get Started</Button>
  </HomeContainer>
 );
}

export default Home;
```

Login.js:

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import styled from "styled-components";
import { FontAwesomeIcon } from "@fortawesome/react-fontawesome";
import { faEye, faEyeSlash } from "@fortawesome/free-solid-svg-icons";
import { faFacebookF, faGoogle, faLinkedinIn } from "@fortawesome/free-brands-svg-icons";
import { signInWithPopup } from "firebase/auth";
import { auth, googleProvider, facebookProvider } from "../firebase";
const Container = styled.div`
  display: flex;
  height: 100vh;
`;

const LeftPanel = styled.div`
  flex: 1;
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  background: white;
  padding: 50px;
`;

const RightPanel = styled.div`
  flex: 1;
  background: linear-gradient(135deg, #16a085, #2ecc71);
  color: white;
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  text-align: center;
`;

const LoginBox = styled.div`
  width: 350px;
  text-align: center;
`;

const Title = styled.h2`
  font-size: 28px;
  font-weight: bold;
  color: #333;
  margin-bottom: 10px;
`;

const Subtitle = styled.p`
  font-size: 14px;
  color: gray;
  margin-bottom: 20px;
`;

const SocialIcons = styled.div`
  display: flex;
  justify-content: center;
  gap: 15px;
  margin-bottom: 15px;
`;
```

```
const SocialButton = styled.button`
  background: #f5f5f5;
  border: none;
  padding: 10px 15px;
  border-radius: 50%;
  cursor: pointer;
  transition: 0.3s;

  &:hover {
    background: #ddd;
  }

  svg {
    font-size: 18px;
    color: #555;
  }
`;

const Divider = styled.div`
  width: 100%;
  text-align: center;
  margin: 10px 0;
  font-size: 12px;
  color: gray;
`;

const InputGroup = styled.div`
  position: relative;
  margin-bottom: 15px;
`;

const Input = styled.input`
  width: 100%;
  padding: 12px;
  padding-right: 40px;
  border: 1px solid #ccc;
  border-radius: 5px;
  outline: none;
  font-size: 14px;
`;

const EyeIcon = styled.span`
  position: absolute;
  right: 10px;
  top: 50%;
  transform: translateY(-50%);
  cursor: pointer;
`;

const Button = styled.button`
  width: 100%;
  padding: 12px;
  background: #16a085;
  color: white;
  font-size: 16px;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  transition: 0.3s;
```

```
  margin-top: 10px;

  &:hover {
    background: #13876c;
  }
`;

const SignupSection = styled.div`
  margin-top: 20px;
  font-size: 14px;

  a {
    color: #16a085;
    text-decoration: none;
    font-weight: bold;
  }
`;

const RightText = styled.h2`
  font-size: 24px;
  font-weight: bold;
`;

const RightDescription = styled.p`
  font-size: 14px;
  margin-top: 10px;
  max-width: 70%;
`;

const SignUpButton = styled.button`
  background: white;
  color: #16a085;
  border: none;
  padding: 10px 20px;
  font-size: 16px;
  border-radius: 25px;
  cursor: pointer;
  margin-top: 15px;
  transition: 0.3s;

  &:hover {
    background: #f5f5f5;
  }
`;

function Login() {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [passwordVisible, setPasswordVisible] = useState(false);
  const navigate = useNavigate();

  const togglePasswordVisibility = () => {
    setPasswordVisible(!passwordVisible);
  };

  const handleLogin = (e) => {
    e.preventDefault();

    const validUsers = [
      { email: "admin@example.com", password: "Admin@123", role: "admin" },
```

```
    { email: "user@example.com", password: "User@123", role: "user" },
    { email: "store@example.com", password: "Store@123", role: "store_owner" }
  ];

  const user = validUsers.find((u) => u.email === email && u.password === password);

  if (user) {
    alert(`Login Successful! Role: ${user.role}`);

    if (user.role === "admin") {
      navigate("/dashboard");
    } else if (user.role === "user") {
      navigate("/stores");
    } else {
      navigate("/store-dashboard");
    }
  } else {
    alert("Invalid Credentials! Please try again.");
  }
};
const handleGoogleLogin = async () => {
  try {
    const result = await signInWithPopup(auth, googleProvider);
    const user = result.user;
    alert(`Google Login Successful! Welcome, ${user.displayName}`);
    navigate("/dashboard"); // Redirect after login
  } catch (error) {
    alert("Google Login Failed: " + error.message);
  }
};

// Facebook Login Function
const handleFacebookLogin = async () => {
  try {
    const result = await signInWithPopup(auth, facebookProvider);
    const user = result.user;
    alert(`Facebook Login Successful! Welcome, ${user.displayName}`);
    navigate("/dashboard"); // Redirect after login
  } catch (error) {
    alert("Facebook Login Failed: " + error.message);
  }
};

return (
  <Container>
    <LeftPanel>
      <LoginBox>
        <Title>Login to Your Account</Title>
        <Subtitle>Login using social networks</Subtitle>
        <SocialIcons>
<SocialButton onClick={handleFacebookLogin}>
  <FontAwesomeIcon icon={faFacebookF} />
</SocialButton>
<SocialButton onClick={handleGoogleLogin}>
  <FontAwesomeIcon icon={faGoogle} />
</SocialButton>
</SocialIcons>

        <Divider>OR</Divider>
        <InputGroup>
```

```
          <Input type="email" placeholder="Email" onChange={(e) => setEmail(e.target.value)} />
        </InputGroup>
        <InputGroup>
          <Input type={passwordVisible ? "text" : "password"} placeholder="Password" onChange={(e)
=> setPassword(e.target.value)} />
          <EyeIcon onClick={togglePasswordVisibility}>
            <FontAwesomeIcon icon={passwordVisible ? faEyeSlash : faEye} />
          </EyeIcon>
        </InputGroup>
        <Button onClick={handleLogin}>Sign In</Button>
        <SignupSection>
          New Here? <a href="/register">Sign Up</a>
        </SignupSection>
      </LoginBox>
    </LeftPanel>
    <RightPanel>
      <RightText>New Here?</RightText>
      <RightDescription>Sign up and discover a great amount of new
opportunities!</RightDescription>
      <SignUpButton onClick={() => navigate("/register")}>Sign Up</SignUpButton>
    </RightPanel>
  </Container>
 );
}

export default Login;
```

Profile.js:

```
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import styled from "styled-components";

const Container = styled.div`
 max-width: 600px;
 margin: 50px auto;
 padding: 20px;
 background: white;
 border-radius: 10px;
 box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
 text-align: center;
`;

const Title = styled.h2`
 margin-bottom: 20px;
`;

const Input = styled.input`
 width: 90%;
 padding: 10px;
 margin: 10px 0;
 border: 1px solid #ccc;
 border-radius: 5px;
`;

const Button = styled.button`
 padding: 10px 15px;
 margin-top: 10px;
 background: #16a085;
 color: white;
```

```
  border: none;
  cursor: pointer;
  border-radius: 5px;
  transition: 0.3s;

  &:hover {
    background: #13876c;
  }
`;

function Profile() {
  const navigate = useNavigate();
  const [profile, setProfile] = useState({
    name: "Venkata Himavanth Gothala",
    email: "2100031924cseh@gmail.com",
    phone: "6302272467",
    education: "B.Tech CSE (AI & Intelligent Process Automation)",
  });

  const [editMode, setEditMode] = useState(false);
  const [editedProfile, setEditedProfile] = useState({ ...profile });

  useEffect(() => {
    const storedProfile = JSON.parse(localStorage.getItem("userProfile"));
    if (storedProfile) setProfile(storedProfile);
  }, []);

  const handleEdit = () => setEditMode(true);
  const handleSave = () => {
    setProfile(editedProfile);
    localStorage.setItem("userProfile", JSON.stringify(editedProfile));
    setEditMode(false);
  };

  return (
    <Container>
      <Title>Profile Information</Title>
      {editMode ? (
        <>
          <Input
            type="text"
            value={editedProfile.name}
            onChange={(e) => setEditedProfile({ ...editedProfile, name: e.target.value })}
          />
          <Input
            type="email"
            value={editedProfile.email}
            onChange={(e) => setEditedProfile({ ...editedProfile, email: e.target.value })}
          />
          <Input
            type="text"
            value={editedProfile.phone}
            onChange={(e) => setEditedProfile({ ...editedProfile, phone: e.target.value })}
          />
          <Input
            type="text"
            value={editedProfile.education}
            onChange={(e) => setEditedProfile({ ...editedProfile, education: e.target.value })}
          />
          <Button onClick={handleSave}>Save</Button>
```

```
      </>
    ) : (
      <>
        <p><strong>Name:</strong> {profile.name}</p>
        <p><strong>Email:</strong> {profile.email}</p>
        <p><strong>Phone:</strong> {profile.phone}</p>
        <p><strong>Education:</strong> {profile.education}</p>
        <Button onClick={handleEdit}>Edit</Button>
      </>
    )}
  </Container>
 );
}

export default Profile;
```

Register.js:

```
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";
import styled from "styled-components";

const RegisterContainer = styled.div`
 display: flex;
 flex-direction: column;
 align-items: center;
 justify-content: center;
 height: 100vh;
`;

const Input = styled.input`
 padding: 10px;
 margin: 10px;
 width: 250px;
`;

const Button = styled.button`
 padding: 10px;
 width: 270px;
 background-color: #007bff;
 color: white;
 cursor: pointer;
`;

function Register() {
 const [name, setName] = useState("");
 const [email, setEmail] = useState("");
 const [address, setAddress] = useState("");
 const [password, setPassword] = useState("");
 const navigate = useNavigate();

 const handleRegister = () => {
  // Add logic to register the user to your backend here
  alert(`Registered: ${name}, ${email}, ${address}`);
  navigate("/login"); // Redirect to login page
 };

 return (
  <RegisterContainer>
   <h2 className="glow">Register</h2>
```

```
      <Input
        type="text"
        placeholder="Name"
        onChange={(e) => setName(e.target.value)}
      />
      <Input
        type="email"
        placeholder="Email"
        onChange={(e) => setEmail(e.target.value)}
      />
      <Input
        type="text"
        placeholder="Address"
        onChange={(e) => setAddress(e.target.value)}
      />
      <Input
        type="password"
        placeholder="Password"
        onChange={(e) => setPassword(e.target.value)}
      />
      <Button onClick={handleRegister}>Register</Button>
    </RegisterContainer>
  );
}

export default Register;
```

StoreDashboard.js:

```
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import styled from "styled-components";

const Container = styled.div`
  max-width: 600px;
  margin: 50px auto;
  padding: 20px;
  background: white;
  border-radius: 10px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
  text-align: center;
`;

const Title = styled.h2`
  margin-bottom: 20px;
`;

const Input = styled.input`
  width: 90%;
  padding: 10px;
  margin: 10px 0;
  border: 1px solid #ccc;
  border-radius: 5px;
`;

const List = styled.ul`
  list-style: none;
  padding: 0;
`;
```

```
const ListItem = styled.li`
  background: #f8f8f8;
  margin: 10px 0;
  padding: 10px;
  border-radius: 5px;
`;

const Stars = styled.div`
  display: flex;
  justify-content: center;
  gap: 5px;
  margin-top: 5px;
`;

const Star = styled.span`
  font-size: 20px;
  color: ${({ active }) => (active ? "#f39c12" : "#ccc")};
`;

const Button = styled.button`
  padding: 10px 15px;
  margin-top: 10px;
  background: #16a085;
  color: white;
  border: none;
  cursor: pointer;
  border-radius: 5px;
  transition: 0.3s;

  &:hover {
    background: #13876c;
  }
`;

function StoreDashboard() {
  const navigate = useNavigate();
  const [ratings, setRatings] = useState([]);
  const [averageRating, setAverageRating] = useState(0);
  const [newPassword, setNewPassword] = useState("");

  useEffect(() => {
    const sampleRatings = [
      { user: "John Doe", rating: 4 },
      { user: "Jane Smith", rating: 5 },
      { user: "Alice Brown", rating: 3 },
      { user: "Michael Scott", rating: 5 },
      { user: "Dwight Schrute", rating: 2 }
    ];

    setRatings(sampleRatings);

    const total = sampleRatings.reduce((sum, r) => sum + r.rating, 0);
    setAverageRating((total / sampleRatings.length).toFixed(1));
  }, []);

  const handleUpdatePassword = () => {
    if (newPassword.length < 8) {
      alert("Password must be at least 8 characters long.");
      return;
    }
```

```
    alert("Password updated successfully!");
    setNewPassword("");
  };

  const handleLogout = () => {
    localStorage.removeItem("isAuthenticated");
    localStorage.removeItem("userRole");
    navigate("/login");
  };

  return (
    <Container>
      <Title>Store Owner Dashboard</Title>
      <h3>Average Store Rating: ⭐ {averageRating}</h3>

      <h4>Users Who Rated Your Store:</h4>
      <List>
        {ratings.map((r, index) => (
          <ListItem key={index}>
            {r.user} -
            <Stars>
              {[1, 2, 3, 4, 5].map((star) => (
                <Star key={star} active={star <= r.rating}>★</Star>
              ))}
            </Stars>
          </ListItem>
        ))}
      </List>

      <h4>Update Password</h4>
      <Input
        type="password"
        placeholder="New Password"
        value={newPassword}
        onChange={(e) => setNewPassword(e.target.value)}
      />
      <Button onClick={handleUpdatePassword}>Update Password</Button>

      <Button onClick={handleLogout}>Logout</Button>
    </Container>
  );
}

export default StoreDashboard;
```

StoreList.js:

```
import React, { useState, useEffect } from "react";
import axios from "axios";

const StoreList = () => {
  const [stores, setStores] = useState([]);
  const [sortField, setSortField] = useState("name");
  const [sortOrder, setSortOrder] = useState("asc");

  useEffect(() => {
    fetchStores();
  }, [sortField, sortOrder]);

  const fetchStores = async () => {
```

```
    try {
      const response = await axios.get(
        `/api/stores?sort=${sortField}&order=${sortOrder}`
      );
      setStores(response.data);
    } catch (error) {
      console.error("Error fetching stores", error);
    }
  };

  const handleSort = (field) => {
    setSortField(field);
    setSortOrder(sortOrder === "asc" ? "desc" : "asc");
  };

  return (
    <div>
      <h2>Store List</h2>
      <table>
        <thead>
          <tr>
            <th onClick={() => handleSort("name")}>Store Name</th>
            <th onClick={() => handleSort("address")}>Address</th>
            <th onClick={() => handleSort("rating")}>Rating</th>
          </tr>
        </thead>
        <tbody>
          {stores.map((store) => (
            <tr key={store._id}>
              <td>{store.name}</td>
              <td>{store.address}</td>
              <td>{store.rating}</td>
            </tr>
          ))}
        </tbody>
      </table>
    </div>
  );
};

export default StoreList;
```

StoreListPage.js:

```
import React, { useState } from "react";
import styled from "styled-components";

const StoreListContainer = styled.div`
  padding: 20px;
`;

const Input = styled.input`
  padding: 10px;
  margin: 10px;
  width: 250px;
`;

const StoreItem = styled.div`
  padding: 10px;
  margin: 10px 0;
```

```
  border: 1px solid #ccc;
  border-radius: 5px;
`;

function StoreList() {
  const [searchTerm, setSearchTerm] = useState("");
  const [stores, setStores] = useState([
    { name: "Store 1", address: "123 Main St" },
    { name: "Store 2", address: "456 Oak St" },
    { name: "Store 3", address: "789 Pine St" },
  ]);

  const handleSearch = () => {
    const filteredStores = stores.filter(
      (store) =>
        store.name.toLowerCase().includes(searchTerm.toLowerCase()) ||
        store.address.toLowerCase().includes(searchTerm.toLowerCase())
    );
    setStores(filteredStores);
  };

  return (
    <StoreListContainer>
      <h2>Registered Stores</h2>
      <Input
        type="text"
        placeholder="Search by name or address"
        onChange={(e) => setSearchTerm(e.target.value)}
      />
      <button onClick={handleSearch}>Search</button>
      {stores.map((store, index) => (
        <StoreItem key={index}>
          <p><strong>Name:</strong> {store.name}</p>
          <p><strong>Address:</strong> {store.address}</p>
        </StoreItem>
      ))}
    </StoreListContainer>
  );
}

export default StoreList;
```

StoreOwnerDashboard.js:

```
import React, { useState, useEffect } from "react";

const StoreOwnerDashboard = ({ storeId }) => {
    const [ratings, setRatings] = useState([]);
    const [averageRating, setAverageRating] = useState(0);

    useEffect(() => {
        const fetchRatings = async () => {
            try {
                const response = await fetch(`/api/stores/${storeId}/ratings`);
                const data = await response.json();
                setRatings(data);

                const totalRating = data.reduce((acc, rating) => acc + rating.rating, 0);
                const avg = data.length > 0 ? totalRating / data.length : 0;
                setAverageRating(avg.toFixed(1));
```

```
        } catch (error) {
          console.error("Error fetching ratings:", error);
        }
      };

      fetchRatings();
    }, [storeId]);

    return (
      <div style={{ padding: "20px" }}>
        <h2>Store Owner Dashboard</h2>
        <h3>Average Rating: ⭐ {averageRating}</h3>

        <table border="1" style={{ width: "100%", marginTop: "20px", textAlign: "left" }}>
          <thead>
            <tr>
              <th>User Name</th>
              <th>Email</th>
              <th>Rating</th>
              <th>Comment</th>
            </tr>
          </thead>
          <tbody>
            {ratings.length > 0 ? (
              ratings.map((rating) => (
                <tr key={rating.id}>
                  <td>{rating.userName}</td>
                  <td>{rating.userEmail}</td>
                  <td>⭐ {rating.rating}</td>
                  <td>{rating.comment}</td>
                </tr>
              ))
            ) : (
              <tr>
                <td colSpan="4" style={{ textAlign: "center" }}>
                  No ratings available.
                </td>
              </tr>
            )}
          </tbody>
        </table>
      </div>
    );
};

export default StoreOwnerDashboard;
```

UserDashboard.js:

```
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import styled from "styled-components";

const Container = styled.div`
  max-width: 600px;
  margin: 50px auto;
  padding: 20px;
  background: white;
  border-radius: 10px;
```

```
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
  text-align: center;
`;

const Title = styled.h2`
  margin-bottom: 20px;
`;

const Input = styled.input`
  width: 90%;
  padding: 10px;
  margin: 10px 0;
  border: 1px solid #ccc;
  border-radius: 5px;
`;

const Button = styled.button`
  padding: 10px 15px;
  margin-top: 10px;
  background: #16a085;
  color: white;
  border: none;
  cursor: pointer;
  border-radius: 5px;
  transition: 0.3s;

  &:hover {
    background: #13876c;
  }
`;

const StoreList = styled.ul`
  list-style: none;
  padding: 0;
`;

const StoreItem = styled.li`
  background: #f8f8f8;
  margin: 10px 0;
  padding: 10px;
  border-radius: 5px;
`;

function UserDashboard() {
  const navigate = useNavigate();
  const [stores, setStores] = useState([]);
  const [storeName, setStoreName] = useState("");
  const [storeAddress, setStoreAddress] = useState("");

  useEffect(() => {
    const savedStores = JSON.parse(localStorage.getItem("stores")) || [];
    setStores(savedStores);
  }, []);

  const saveStores = (updatedStores) => {
    localStorage.setItem("stores", JSON.stringify(updatedStores));
    setStores(updatedStores);
  };

  const addStore = () => {
```

```
    if (storeName.length < 3 || storeAddress.length < 5) {
      alert("Store Name must be at least 3 characters & Address at least 5.");
      return;
    }

    const newStore = { name: storeName, address: storeAddress, rating: 0 };
    saveStores([...stores, newStore]);
    setStoreName("");
    setStoreAddress("");
  };

  const handleLogout = () => {
    localStorage.removeItem("isAuthenticated");
    localStorage.removeItem("userRole");
    navigate("/login");
  };

  return (
    <Container>
      <Title>User Dashboard</Title>
      <h3>Add a New Store</h3>
      <Input
        type="text"
        placeholder="Store Name"
        value={storeName}
        onChange={(e) => setStoreName(e.target.value)}
      />
      <Input
        type="text"
        placeholder="Store Address"
        value={storeAddress}
        onChange={(e) => setStoreAddress(e.target.value)}
      />
      <Button onClick={addStore}>Add Store</Button>

      <h3>All Stores</h3>
      <StoreList>
        {stores.map((store, index) => (
          <StoreItem key={index}>
            <strong>{store.name}</strong> - {store.address}
          </StoreItem>
        ))}
      </StoreList>

      <Button onClick={handleLogout}>Logout</Button>
    </Container>
  );
}

export default UserDashboard;
```

GlobalStyle.js:

```
import { createGlobalStyle } from "styled-components";

const GlobalStyle = createGlobalStyle`
  * {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
```

```
    font-family: 'Poppins', sans-serif;
  }

  body {
    background: url('https://e1.pxfuel.com/desktop-wallpaper/44/391/desktop-wallpaper-supermarket-high-quality-general-store.jpg') no-repeat center center fixed;
    background-size: cover;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
  }

  .glow {
    text-shadow: 0 0 20px #0ff, 0 0 40px #0ff, 0 0 60px #0ff;
  }
`;

export default GlobalStyle;
```

App.js:

```
import React from "react";
import { BrowserRouter as Router, Route, Routes } from "react-router-dom";
import GlobalStyle from "./styles/GlobalStyle";
import Home from "./pages/Home";
import Login from "./pages/Login";
import Register from "./pages/Register";
import Dashboard from "./pages/Dashboard";
import StoreList from "./pages/StoreList";
import UpdatePassword from './components/UpdatePassword';
import Profile from "./components/Profile";

function App() {
  return (
    <Router>
      <GlobalStyle />
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/login" element={<Login />} />
        <Route path="/register" element={<Register />} />
        <Route path="/dashboard" element={<Dashboard />} />
        <Route path="/stores" element={<StoreList />} />
        <Route path="/profile" element={<Profile />} />
        <Route path="/update-password" element={<UpdatePassword />} />
        <Route path="/stores" element={<StoreList />} />
        <Route path="/profile" element={<Profile />} />


      </Routes>
    </Router>
  );
}

export default App;
```

Firebase.js:

```
import { initializeApp } from "firebase/app";
import { getAuth, GoogleAuthProvider, FacebookAuthProvider } from "firebase/auth";
```

```
const firebaseConfig = {
  apiKey: "YOUR_FIREBASE_API_KEY",
  authDomain: "YOUR_FIREBASE_AUTH_DOMAIN",
  projectId: "YOUR_FIREBASE_PROJECT_ID",
  storageBucket: "YOUR_FIREBASE_STORAGE_BUCKET",
  messagingSenderId: "YOUR_FIREBASE_MESSAGING_SENDER_ID",
  appId: "YOUR_FIREBASE_APP_ID"
};

const app = initializeApp(firebaseConfig);
const auth = getAuth(app);
const googleProvider = new GoogleAuthProvider();
const facebookProvider = new FacebookAuthProvider();

export { auth, googleProvider, facebookProvider };
```

Index.js:

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);

reportWebVitals();
```

Backend Implementation:

config(folder):

db.js:

```
const { Sequelize } = require("sequelize");
require("dotenv").config();

const sequelize = new Sequelize(process.env.DB_NAME, process.env.DB_USER,
process.env.DB_PASS, {
  host: process.env.DB_HOST,
  dialect: process.env.DB_DIALECT,
  logging: false,
});
```

```
sequelize
  .authenticate()
  .then(() => console.log("✅ Database Connected"))
  .catch((err) => console.error("❌ Database Connection Failed:", err.message));

module.exports = sequelize;
```

Test_db.js:

```
const mysql = require("mysql2");

const connection = mysql.createConnection({
  host: "127.0.0.1",
  user: "root",
  password: "yourpassword",
  database: "store_db"
});

connection.connect((err) => {
  if (err) {
    console.error("❌ MySQL Connection Failed:", err.message);
  } else {
    console.log("✅ MySQL Connected Successfully!");
  }
  connection.end();
});
```

Middleware(folder):

authMiddleware.js:

```
const jwt = require("jsonwebtoken");

module.exports = (req, res, next) => {
  const token = req.header("Authorization");
  if (!token) return res.status(401).json({ message: "Access Denied" });

  try {
    const verified = jwt.verify(token, process.env.JWT_SECRET);
    req.user = verified;
    next();
  } catch (err) {
    res.status(400).json({ message: "Invalid Token" });
  }
};
```

Models(folder):

Rating.js:

```
const { DataTypes } = require("sequelize");
const sequelize = require("../config/db");
const User = require("./User");
const Store = require("./Store");

const Rating = sequelize.define("Rating", {
  userId: { type: DataTypes.INTEGER, references: { model: User, key: "id" } },
  storeId: { type: DataTypes.INTEGER, references: { model: Store, key: "id" } },
  rating: { type: DataTypes.INTEGER, allowNull: false }
});
```

```
module.exports = Rating;
```

Store.js:

```
const { DataTypes } = require("sequelize");
const sequelize = require("../config/db");

const Store = sequelize.define("Store", {
  name: { type: DataTypes.STRING, allowNull: false },
  email: { type: DataTypes.STRING, allowNull: false, unique: true },
  address: { type: DataTypes.STRING(400), allowNull: false },
  rating: { type: DataTypes.FLOAT, defaultValue: 0 }
});

module.exports = Store;
```

User.js:

```
const mongoose = require("mongoose");

const userSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true,
    minlength: 20,
    maxlength: 60,
  },
  email: {
    type: String,
    required: true,
    unique: true,
    match: /^[^\s@]+@[^\s@]+\.[^\s@]+$/,
  },
  address: {
    type: String,
    required: true,
    maxlength: 400,
  },
  password: {
    type: String,
    required: true,
  },
  role: {
    type: String,
    enum: ["System Administrator", "Normal User", "Store Owner"],
    required: true,
  },
});

module.exports = mongoose.model("User", userSchema);
```

Routes(folder):

authRoutes.js:

```
const express = require("express");
const bcrypt = require("bcryptjs");
const jwt = require("jsonwebtoken");
const User = require("../models/User");
```

```
const router = express.Router();

const validatePassword = (password) => {
  const regex = /^(?=.*[A-Z])(?=.*\W).{8,16}$/;
  return regex.test(password);
};

router.post("/register", async (req, res) => {
  try {
    const { name, email, address, password, role } = req.body;

    if (!name || !email || !address || !password || !role) {
      return res.status(400).json({ error: "All fields are required." });
    }

    if (name.length < 20 || name.length > 60) {
      return res.status(400).json({
        error: "Name must be between 20 and 60 characters.",
      });
    }

    if (address.length > 400) {
      return res.status(400).json({
        error: "Address cannot exceed 400 characters.",
      });
    }

    if (!validatePassword(password)) {
      return res.status(400).json({
        error:
          "Password must be 8-16 characters long, include at least 1 uppercase letter and 1 special
character.",
      });
    }

    const existingUser = await User.findOne({ email });
    if (existingUser) {
      return res.status(400).json({ error: "Email already in use." });
    }

    const salt = await bcrypt.genSalt(10);
    const hashedPassword = await bcrypt.hash(password, salt);

    const newUser = new User({
      name,
      email,
      address,
      password: hashedPassword,
      role,
    });

    await newUser.save();

    res.status(201).json({ message: "User registered successfully!" });
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: "Server error. Please try again." });
  }
});
```

```
router.post("/login", async (req, res) => {
  try {
    const { email, password } = req.body;

    const user = await User.findOne({ email });
    if (!user) {
      return res.status(400).json({ error: "Invalid email or password." });
    }

    const isMatch = await bcrypt.compare(password, user.password);
    if (!isMatch) {
      return res.status(400).json({ error: "Invalid email or password." });
    }

    const token = jwt.sign({ id: user._id, role: user.role }, "your_jwt_secret", {
      expiresIn: "1h",
    });

    res.status(200).json({ token, role: user.role, message: "Login successful!" });
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: "Server error. Please try again." });
  }
});

module.exports = router;
```

ratingRoutes.js:

```
const express = require("express");
const Rating = require("../models/Rating");
const Store = require("../models/Store");
const authMiddleware = require("../middleware/authMiddleware");

const router = express.Router();

router.post("/", authMiddleware, async (req, res) => {
  if (req.user.role !== "user") {
    return res.status(403).json({ message: "Only normal users can submit ratings" });
  }

  const { storeId, rating } = req.body;

  if (!storeId || rating < 1 || rating > 5) {
    return res.status(400).json({ message: "Invalid store ID or rating must be between 1 and 5" });
  }

  try {
    let userRating = await Rating.findOne({ where: { userId: req.user.id, storeId } });

    if (userRating) {
      userRating.rating = rating;
      await userRating.save();
      return res.json({ message: "Rating updated successfully", userRating });
    }

    userRating = await Rating.create({ userId: req.user.id, storeId, rating });

    const ratings = await Rating.findAll({ where: { storeId } });
```

```
    const avgRating = ratings.reduce((acc, r) => acc + r.rating, 0) / ratings.length;

    const store = await Store.findByPk(storeId);
    store.rating = avgRating.toFixed(1);
    await store.save();

    res.json({ message: "Rating submitted successfully", userRating, avgRating });
  } catch (error) {
    res.status(500).json({ message: "Server error", error });
  }
});

router.get("/:storeId", async (req, res) => {
  const { storeId } = req.params;

  try {
    const ratings = await Rating.findAll({ where: { storeId } });
    res.json(ratings);
  } catch (error) {
    res.status(500).json({ message: "Server error", error });
  }
});

router.get("/user/:storeId", authMiddleware, async (req, res) => {
  const { storeId } = req.params;

  try {
    const rating = await Rating.findOne({ where: { storeId, userId: req.user.id } });
    if (!rating) {
      return res.status(404).json({ message: "No rating found for this store" });
    }
    res.json(rating);
  } catch (error) {
    res.status(500).json({ message: "Server error", error });
  }
});

module.exports = router;
```

storeRoutes.js:

```
const express = require("express");
const Store = require("../models/Store");
const router = express.Router();

router.get("/stores", async (req, res) => {
  try {
    let query = {};
    if (req.query.name) {
      query.name = { $regex: req.query.name, $options: "i" };
    }
    if (req.query.address) {
      query.address = { $regex: req.query.address, $options: "i" };
    }

    const stores = await Store.find(query);
    res.status(200).json(stores);
  } catch (error) {
    res.status(500).json({ error: "Error fetching stores" });
  }
```

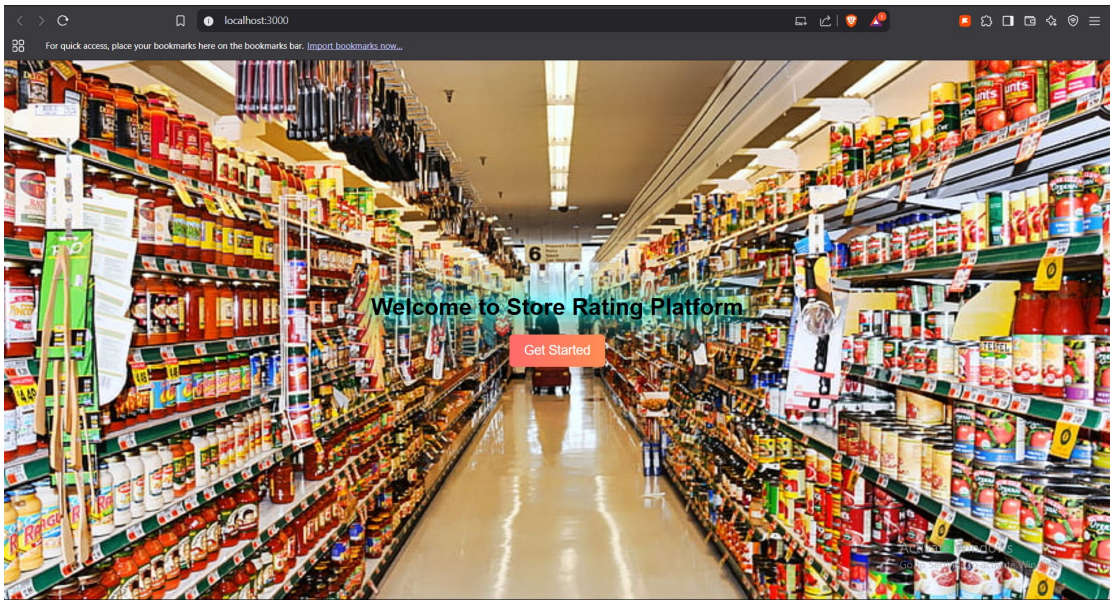
});

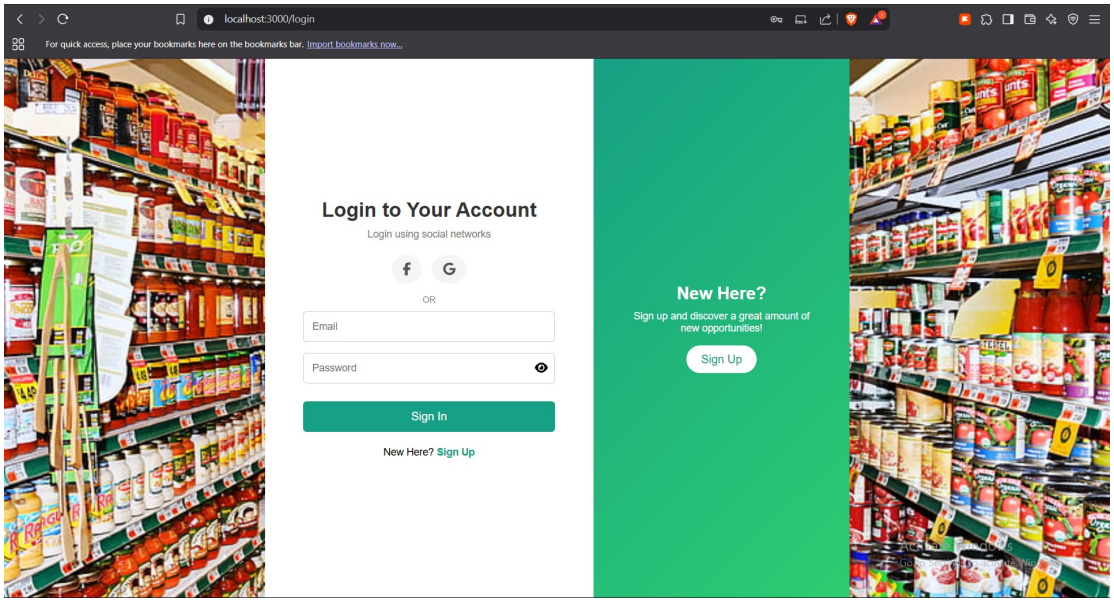
module.exports = router;

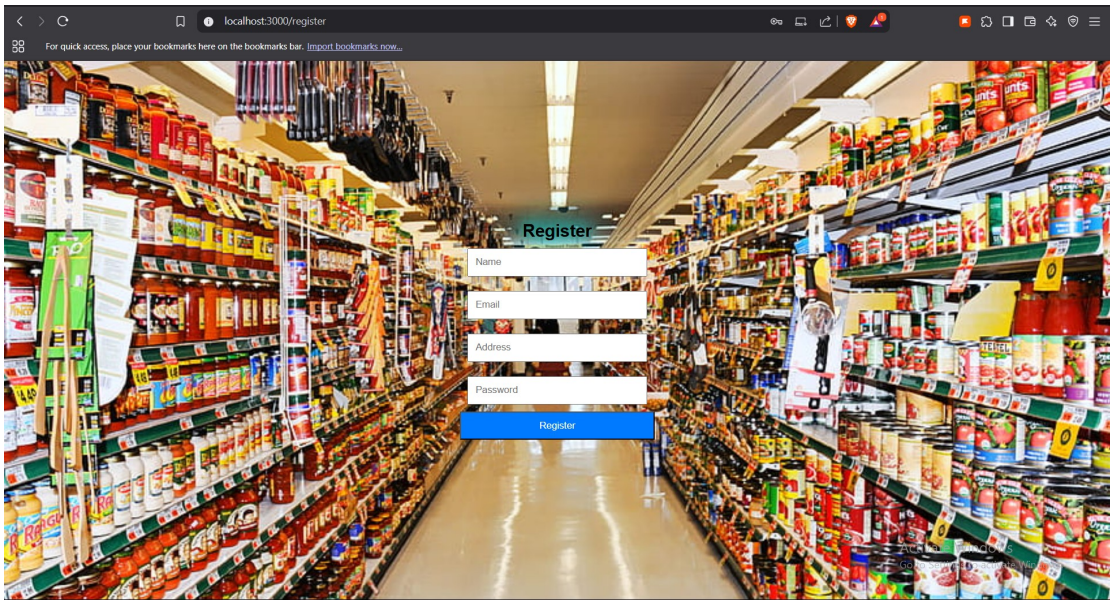
Server.js:

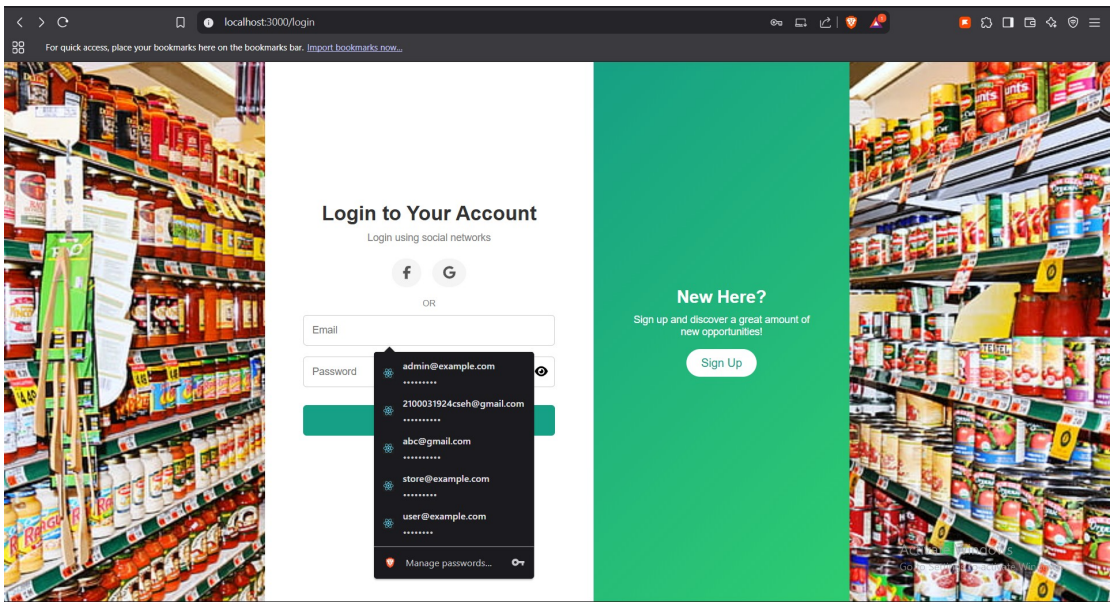
```
require("dotenv").config();
const express = require("express");
const cors = require("cors");
const sequelize = require("./config/db");

const authRoutes = require("./routes/authRoutes");
const storeRoutes = require("./routes/storeRoutes");
const ratingRoutes = require("./routes/ratingRoutes");

const app = express();
app.use(express.json());
app.use(cors());

app.use("/api/auth", authRoutes);
app.use("/api/stores", storeRoutes);
app.use("/api/ratings", ratingRoutes);

sequelize.sync().then(() => {
  console.log("✅ Database Synced");
  app.listen(5000, () => console.log("🚀 Server running on port 5000"));
});
```

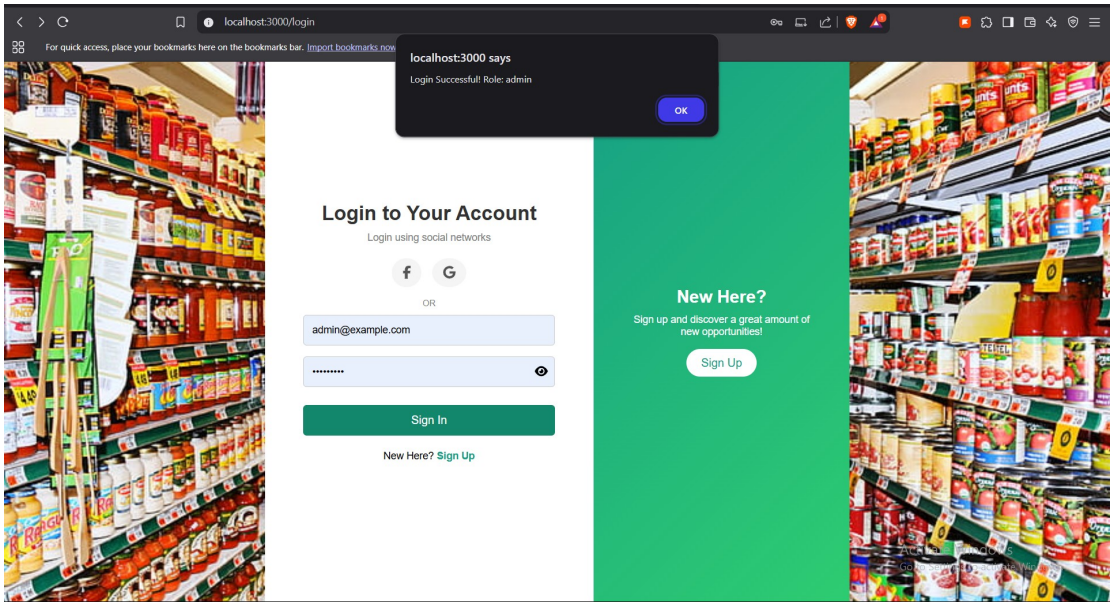
Outputs:

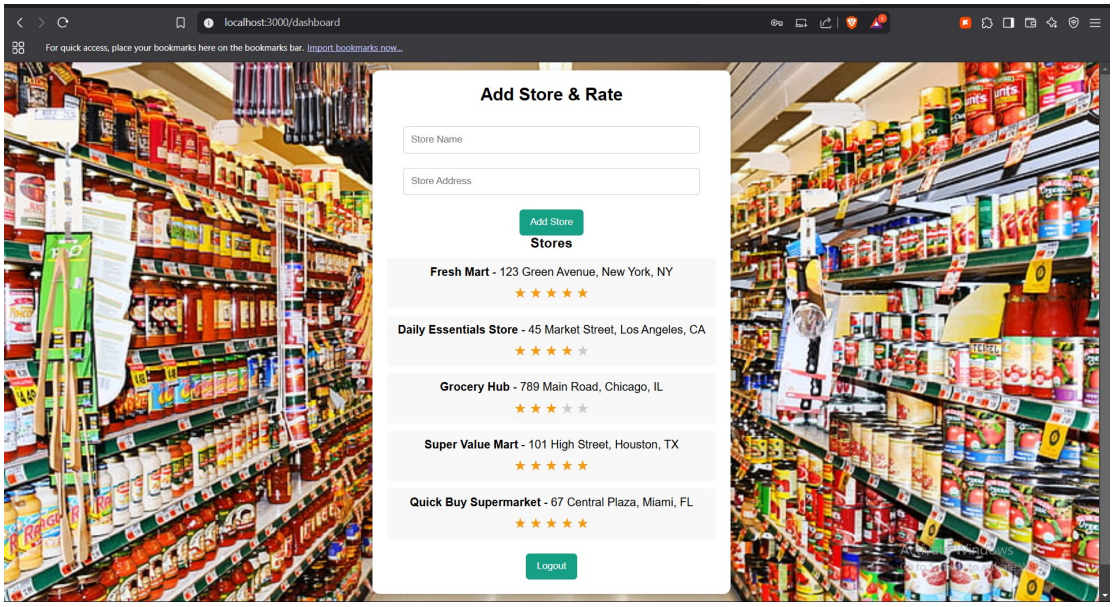



localhost:3000/login
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Login to Your Account
Login using social networks
OR
Email
Password
Sign In
New Here? Sign Up
New Here?
Sign up and discover a great amount of new opportunities!
Sign Up


localhost:3000/register
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Register
Name
Email
Address
Password
Register


localhost:3000/login
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Login to Your Account
Login using social networks
OR
Email
Password
admin@example.com
2100031924cseh@gmail.com
abc@gmail.com
store@example.com
user@example.com
Manage passwords...
New Here?
Sign up and discover a great amount of new opportunities!
Sign Up


localhost:3000/login
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
localhost:3000 says
Login Successful! Role: admin
OK
Login to Your Account
Login using social networks
OR
admin@example.com
Sign In
New Here? Sign Up
New Here?
Sign up and discover a great amount of new opportunities!
Sign Up


localhost:3000/dashboard
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Add Store & Rate
Store Name
Store Address
Add Store
Stores
Fresh Mart - 123 Green Avenue, New York, NY
Daily Essentials Store - 45 Market Street, Los Angeles, CA
Grocery Hub - 789 Main Road, Chicago, IL
Super Value Mart - 101 High Street, Houston, TX
Quick Buy Supermarket - 67 Central Plaza, Miami, FL
Logout